COMP 2501A - Computer Game Design and Development
Winter 2024

Carleton University
School of Computer Science
Course Outline
Last update: December 21, 2023

Course Information

Instructor: Oliver van Kaick
Contact: Oliver.vanKaick at carleton.ca
Classroom: Please check the public class schedule
Lectures: Mondays and Wednesdays, 2:35pm – 3:55pm
Tutorials: Check your schedule on Carleton Central
Student hours: Information on student hours can be found in Brightspace
Course Website: https://brightspace.carleton.ca/d2l/home/220950

For information about Carleton's academic year, including registration and withdrawal dates, see Carleton's Academic Calendar.

Teaching Assistants

A list of teaching assistants and their contact/office hours information will be posted to Brightspace once the course starts.

Course Summary

Introduction to the practical development of computer games. The course covers a variety of mathematical concepts, algorithms, and software technologies relevant for the creation of 2D games. The course includes programming assignments.

Course Calendar Description

Introduction to the practical development of computer games and engine architecture. Topics include: vector and matrix operations; coordinate systems and transformations; physical simulation; collision detection; AI; path planning; hardware-accelerated real-time rendering. Special attention is given to implementation of real-time rendering in a low-level language.
Includes: Experiential Learning Activity
Prerequisite(s): COMP 1501, COMP 2401 with a minimum grade of C-, and MATH 1104.
Lectures three hours a week, tutorial one and a half hours a week.

Topics Covered

- Game architecture: MVC design pattern, game object management
- Mathematical foundations: vector operations, coordinate systems, and transformations
- Introduction to hardware-accelerated real-time rendering: geometry and shaders
- Introduction to OpenGL
- Physical simulation and collision detection
- Game AI and path planning

Learning Outcomes

At the end of this course, students will be able to:

- Summarize the main components necessary for the development of a computer game based on 2D graphics and physical simulation.
- Explain the principles behind the fundamental techniques used for the creation of 2D games (the topics listed above), discussing the mathematical operations and algorithms involved in these techniques.
- Identify the most suitable techniques to create specific features in a 2D computer game.
- Implement a basic 2D game in C++ with OpenGL graphics and auxiliary libraries.

Resources

We do not have an assigned textbook for the course. I recommend using Sanjay Madhav's *Game Programming: Algorithms and Techniques* as a reference for reviewing the different topics covered in the lectures. For the assignments and the course project, we will be programming in C++ and using a set of libraries that build on OpenGL. The libraries will be posted on Brightspace. For detailed questions on programming with C++ and OpenGL, there are a wealth of books, websites, and online tutorials that provide information; a few recommendations are provided in the Brightspace page.

Computer Requirement

For the programming assignments, you will need a computer that has a GPU suitable to run computer games using the OpenGL library. Recent Windows or Linux computers with GPU support should work fine. MacOS has discontinued support for OpenGL.

Assessment Scheme

Grading scheme (the specific deadlines can be found in Brightspace):

- Assignments: 35%, approximately every two weeks.
- In-class exams: 15%, around February and March.
- Course project: 15%, due at the end of classes.
- Final exam: 35%, scheduled centrally, during exam week.

Note that you need to obtain a passing grade in the (in-class exams + final) to pass the course.

Late Assignment Policy

Assignment deadlines are strict. The following scheme is applied to late submissions (which includes assignments and the final course project):
- 3 hours late: no penalty
- 3 to 12 hours late: -10%
- 12 to 24 hours late: -20%
- More than one day late: assignment receives a grade of zero.

Assignment submissions are handled electronically (i.e., through Brightspace). Technical problems do not exempt you from submitting on time. So, if you wait until the last minute and then have issues with your connection, you will receive a deduction according to the scheme above. Consequently, you are advised to:

- Periodically upload you progress (e.g., upload your progress to Brightspace after each major change; we will only grade your last submission).
- Submit your final work at least one hour in advance of the due date and time.
- Store backups of your assignments in the cloud, e.g., OneDrive, Dropbox, a private GitHub repository. However, your assignment has to be submitted to Brightspace so that we have a timestamped submission. Urls to the cloud will not be accepted.

The assignments consist of programming tasks. If any of the assignments that you submit does not compile or run, it will receive a mark of zero. Consequently, after you upload your submission to Brightspace, you should re-download it immediately and ensure that the project can be created with cmake, compiled, and run.

You are expected to demonstrate good programming practices at all times and your code may be penalized if it is poorly written. You are also expected to do the necessary preparatory work (i.e., devising an algorithm) before you start coding.

Academic Integrity and code reuse

If you are unsure about the expectations regarding academic integrity (how to use and cite references, how much collaboration with lab- or classmates is appropriate), ask your instructor. Sharing assignment specifications or posting them online (to sites like Chegg, CourseHero, OneClass, etc.) is considered academic misconduct. You are never permitted to post, share, or upload course materials without explicit permission from your instructor. Academic integrity offences are reported to the office of the Dean of Science. Information, the process, and penalties for such offences can be found on the ODS webpage: https://science.carleton.ca/students/academic-integrity/.

You are free to make use of art assets found online provided that their license allows you to freely use the assets and you credit the source. Code fragments that are not of your own authorship are allowed under the following conditions: 1. The code should not be implementing the main tasks required for an assignment, but rather serve for adding additional features to the project; 2. Provide credit to the original author of the code and make sure that you understand what the code is doing.

Use of tools based on artificial intelligence

All of the assessed activities in this course except for the project were designed to be completed by an individual working alone. The use of artificial intelligence tools such as ChatGPT, Copilot, etc., is allowed in programming assignments. However, if you use any of these tools, you have to disclose this in a radme.txt file or report when submitting the activity, just as you would do when reusing code from other sources. Explain what portions of the assignment were created solely by you and what portions were created with the aid of the tool. The use of any of these tools without disclosure will be considered academic misconduct. An exception to this rule is made for automated grammar and punctuation checking tools (such as Grammarly). In addition, do not fully trust these tools and double check any generated code, as AI tools are known to produce incorrect outputs.

Undergraduate Academic Advisors

The Undergraduate Advisors for the School of Computer Science are available in Room 5302HP; or by email at scs.ug.advisor@cunet.carleton.ca. The undergraduate advisors can assist with information about prerequisites and preclusions, course substitutions/equivalencies, understanding your academic audit and the remaining requirements for graduation. The undergraduate advisors will also refer students to appropriate resources such as the Science Student Success Centre, Learning Support Services and Writing Tutorial Services.

SCS Computer Laboratory

Students taking a COMP course can access the SCS computer labs. The lab schedule and location can be found at: https://carleton.ca/scs/tech-support/computer-laboratories/. All SCS computer lab and technical support information can be found at: https://carleton.ca/scs/tech-support/. Technical support staff may be contacted in-person or virtually, see this page for details: https://carleton.ca/scs/tech-support/contact-it-support/.

University Policies:

Academic Accommodations: Carleton is committed to providing academic accessibility for all individuals. Please review the academic accommodation available to students here: https://students.carleton.ca/course-outline/.

Student Academic Integrity Policy. Every student should be familiar with the Carleton University Student Academic Integrity policy. A student found in violation of academic integrity standards may be sanctioned with penalties which range from a reprimand to receiving a grade of F in the course, or even being suspended or expelled from the University. Examples of punishable offences include plagiarism and unauthorized collaboration. Any such reported offences will be reviewed by the office of the Dean of Science. More information on this policy may be found on the ODS Academic Integrity page: https://carleton.ca/registrar/academic-integrity/.

Plagiarism. As defined by Senate, "plagiarism is presenting, whether intentional or not, the ideas, expression of ideas or work of others as one's own". Such reported offences will be reviewed by the office of the Dean of Science. More information and standard sanction guidelines can be found here: https://science.carleton.ca/students/academic-integrity/.

Unauthorized Collaboration. Senate policy states that "to ensure fairness and equity in assessment of term work, students shall not co-operate or collaborate in the completion of an academic assignment, in whole or in part, when the instructor has indicated that the assignment is to be completed on an individual basis".